



Salsa: Beyond Model Checking*

Ramesh Bharadwaj

Center for High Assurance Computer Systems

Naval Research Laboratory

Washington, DC 20375 USA

ramesh@itd.nrl.navy.mil

Salsa web site: www.reactive-systems.com/salsa

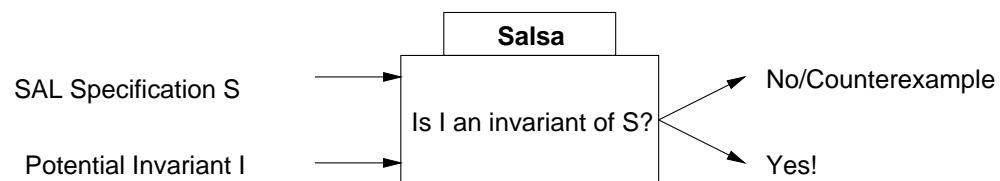
* Work Supported by the Office of Naval Research.

Report Documentation Page				Form Approved OMB No. 0704-0188	
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
1. REPORT DATE 2001		2. REPORT TYPE		3. DATES COVERED 00-00-2001 to 00-00-2001	
4. TITLE AND SUBTITLE Salsa: Beyond Model Checking				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Research Laboratory, Center for High Assurance Computer Systems, 4555 Overlook Avenue, SW, Washington, DC, 20375				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited					
13. SUPPLEMENTARY NOTES The original document contains color images.					
14. ABSTRACT					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES 15	19a. NAME OF RESPONSIBLE PERSON
a. REPORT unclassified	b. ABSTRACT unclassified	c. THIS PAGE unclassified			

What does Salsa do?



2



Why does Salsa work?



3

Goals:

- As easy to use as a model checker
- Scales like a theorem prover

Practical usage of Salsa:

- NRL: Cryptographic Device (CD) [27]
- Reactive-Systems Inc/Ford: Simulink/Stateflow specifications
- SUNY Stony Brook: CAN bus protocol

The Verification Problem

Given the following:

- A system description
- A set of environmental assumptions
- A set of required properties (one-state or two-state)

Verification is the process of:

- Extracting **models** from the system description.
Sufficient to establish the properties of interest.
- Applying a verification tool to the model to verify/refute properties.

Very likely that a property is not provable (or wrong).

A tool should provide **diagnostic information**.

How can a tool help?



5

1. **Diagnostic information**
Counterexamples.
2. **Comprehensible diagnostics**
In the “language” of the original description.
3. **Compact diagnostics**
4. **No misdiagnoses**
Very hard to achieve in practice.

Conventional Wisdom



6

Model Checking

- Automatic, easy to use, counterexamples.

Theorem Proving

- Too general, too expensive, hard to use,...

But the reality is...

Attributes of Model Checking:

1. Completeness
2. Termination
3. Diagnostic information
4. State explosion problem

Attributes of Theorem Proving:

1. Incompleteness
Auxiliary lemmas.
2. Not guaranteed to terminate
Decision procedures.
3. Diagnostic information?
Make it comprehensible to layfolk.
4. Infinite State

Combines Model Checking and Theorem Proving

Strengths:

- Reliance on *decision procedures*
- Combination of decision procedures
- Guaranteed termination
- Counterexamples
- Push-button automation

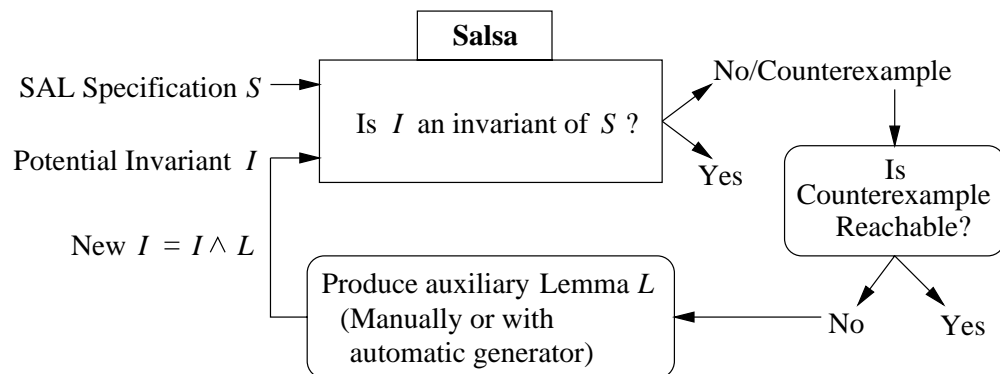
Weaknesses:

- Counterexample not a trace
- Incomplete – counterexamples must be validated

Process for using Salsa



9



Salsa vs. Model Checking



10

Salsa...

- Can handle specs too large for model checkers
(single pre-image vs fixed point computation)
- **More automatic!**
(no manual abstractions)
- Counterexamples

- CCC of the SCR Toolset
(tautology checking vs UC) [22,23,24]
- TAME/PVS [3,30]
- InVeSt [5,6]
- Graf's tools [21,32]
- STeP [11]
- SPIN and SMV on software specs [2,9,16,22]

Notes:

- First four designed for ease-of use.
- First three provide counterexamples.
- STeP requires user interaction.
- Model checkers require the application of abstraction [2,7,8,9,10,27]; they may not always scale (i.e., neither verify/refute) [27].

Empirical Results



12

<i>Specification</i>	<i>Number of</i>			<i>Time (in S) to Check Disj</i>		<i>Number of Failed VCs</i>	
	<i>VCs</i>	<i>BDD Vars</i>	<i>Constrs</i>	<i>SCR Tool</i>	<i>Salsa</i>	<i>SCR Tool</i>	<i>Salsa</i>

Specifications containing mostly booleans and enumerated types

safety-injection	13	16	3	0.5	0.2	0	0
bomb-release-1	12	34	9	0.4	0.2	0	0
a7-modes	6171	158	3	145.9	68.9	110	152

Specifications containing mostly numerical variables

autopilot	29	50	27	1.5	1.0	0	0
home-heating	98	112	55	∞_t	4.8	n.a.	0
cruise-control	123	114	75	21.0	3.6	6	3
navy-1	252	115	78	322.8	59.7	0	0
navy	397	147	102	390.1	198.2	0	0
bomb-release-2	339	319	230	∞_t	246.0	n.a.	11
wcp	58	611	104	-	77.4	-	0

Empirical Results (con't)



13

Spec.	Props. or #	Time (in seconds)				Props. True?	Aux. Lemmas?
		Salsa	SPIN	SMV	TAME		
sis	4	0.8	36.0	155.0	68	Yes	Yes
bre11	2	1.3	∞_t	∞_t	30	Yes	No
autop	2	1.5	∞_t	∞_t	82	Yes	No
navy	7	396.0	∞_t	-	874	Yes	Yes
wcp	# 303	295.4	∞_t	-	∞_t	No	No
	# 304	923.3*	∞_t	-	19	No	No
	# 305	2.4	∞_t	-	8	No	No

- Consistency checking goals achieved
 - Faster
 - Handles integers
 - Able to handle bigger specs
- Bonus: also handles user properties.
 - Handles specs too big for model checkers
 - Seems to be “in the ball park” with PVS
- Weaknesses (w.r.t model checkers):
 - Incompleteness
 - Two-state counterexamples rather than trace from start state.

Stages of Acceptance of Innovation



15

An innovation has three stages of acceptance: First, it is dismissed as rubbish, then it's merely nonviable, and finally it's obvious and trivial – “What we've done all along.”

John Vlissides.